Architect and Build an End-to-End AWS Web Application

Building a Web Application that calculates the exponent(Power) for any Base(Number) provided.

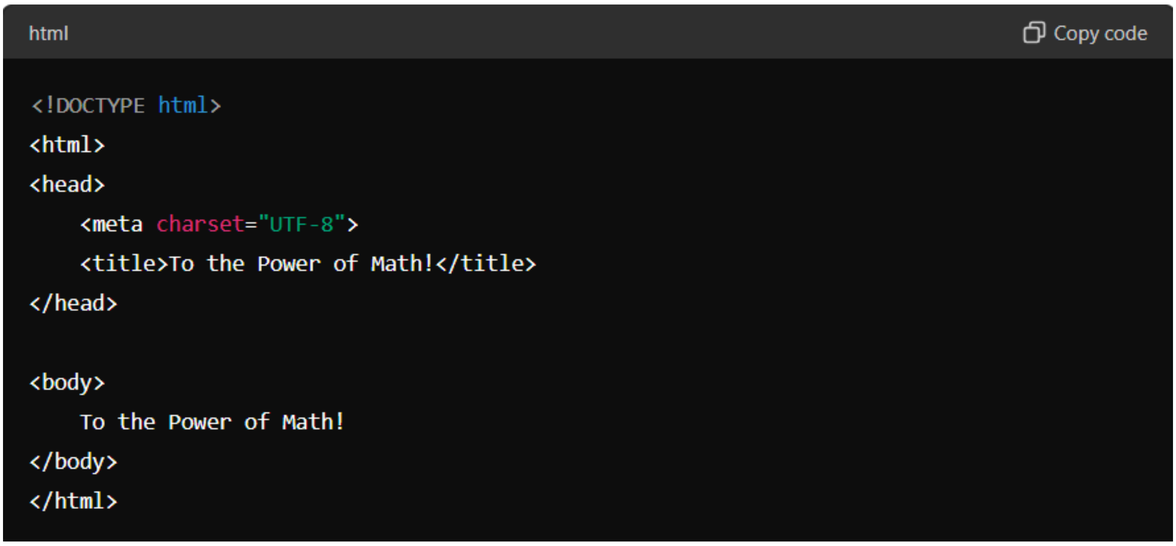
FRONT END:

- **Editor Tool:** VS Code
- **Languages:** HTML, JSON, Python, CSS

SERVICES :

- **AWS Amplify**
- **AWS Lambda**
- **Amazon API Gateway**
- **Amazon Dyanamo DB**
- **AWS Identity and Access Management**

Step 1: Write a HTML Language code to structure and present content on the web,



```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>To the Power of Math!</title>
</head>

<body>
    To the Power of Math!
</body>
</html>
```

Save the file as Index.html and Zip it.

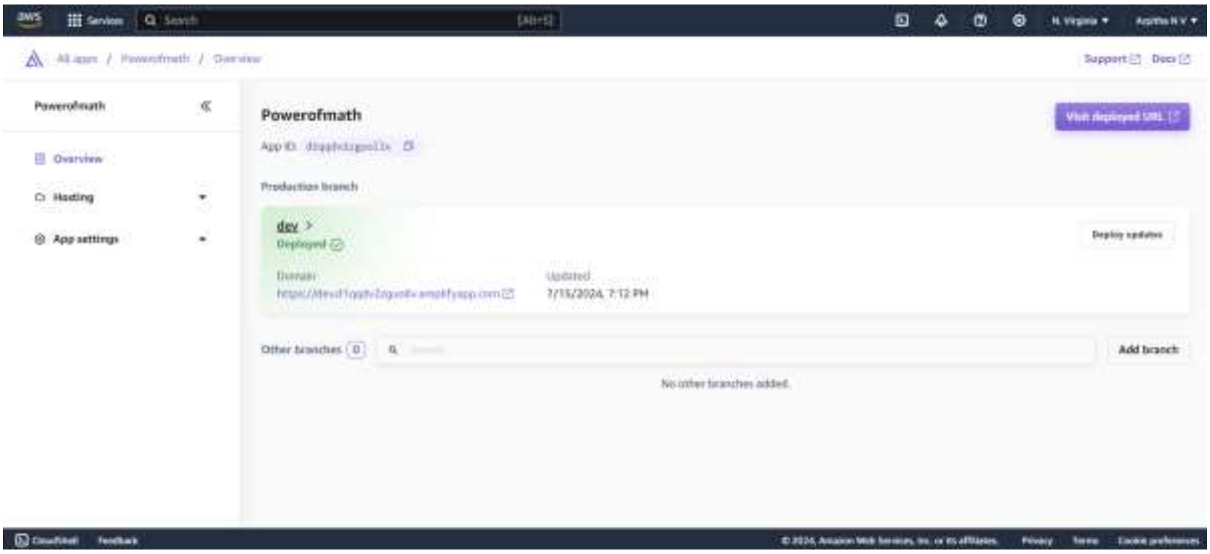
Step 2: Open AWS Management Console page and navigate to AWS Amplify.

Create a New App without a Git Provider option.

Provide a name for your application – Powerofmath

Drag a Zip File that was create earlier, Save and Deploy

Once the Deployment is Successful.



Powerofmath

App ID: d1qgdvdzgvo11x

Production branch

dev >
Deployed ✓

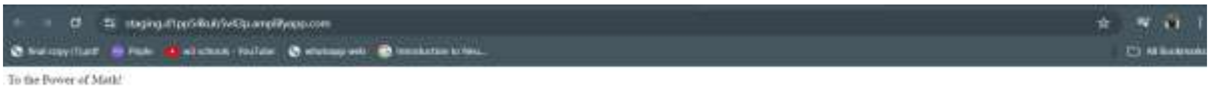
Domain: https://dev.d1qgdv2zgvo11x.amplifyapp.com

Updated: 7/15/2024, 7:12 PM

Other branches (0)

No other branches added.

Copy the URL link provided in the Domain and paste in a new tab,

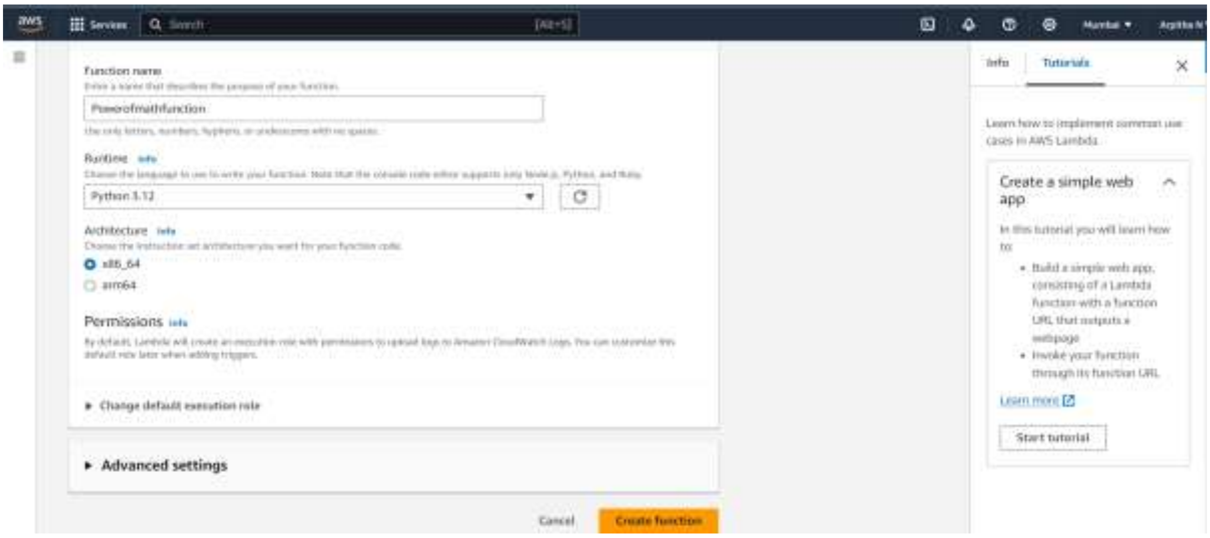


To the Power of Math!

Step 3: Create a Lambda Function (Code that run upon some trigger)

Go to the Management console Page – Functions – Create Function -- Author From Scratch

Select the Latest Version of Python.



Function name
Enter a name that describes the purpose of your function.
Powerofmathfunction
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.12

Architecture Info
Choose the instruction set architecture you want for your function code.
● x86_64
○ arm64

Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

Learn how to implement common use cases in AWS Lambda.

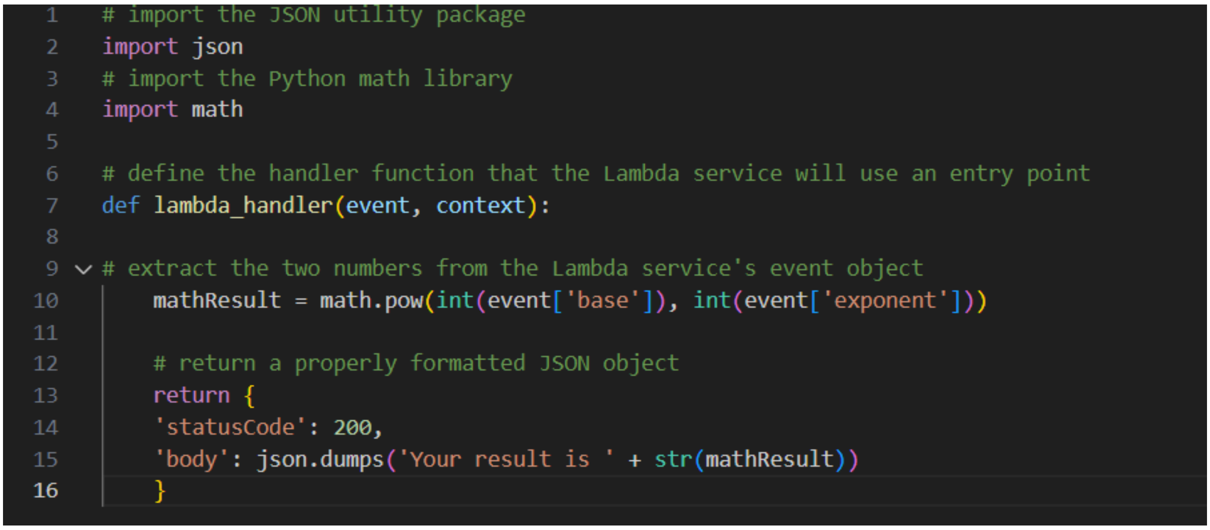
Create a simple web app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

Click on Create Function.

Write a Python Code which defines an AWS Lambda function that computes the power of a given base and exponent passed in the event object. It returns the result as a JSON response.



```python
# import the JSON utility package
import json
# import the Python math library
import math

# define the handler function that the Lambda service will use an entry point
def lambda_handler(event, context):

# extract the two numbers from the Lambda service's event object
    mathResult = math.pow(int(event['base']), int(event['exponent']))

    # return a properly formatted JSON object
    return {
    'statusCode': 200,
    'body': json.dumps('Your result is ' + str(mathResult))
    }
```

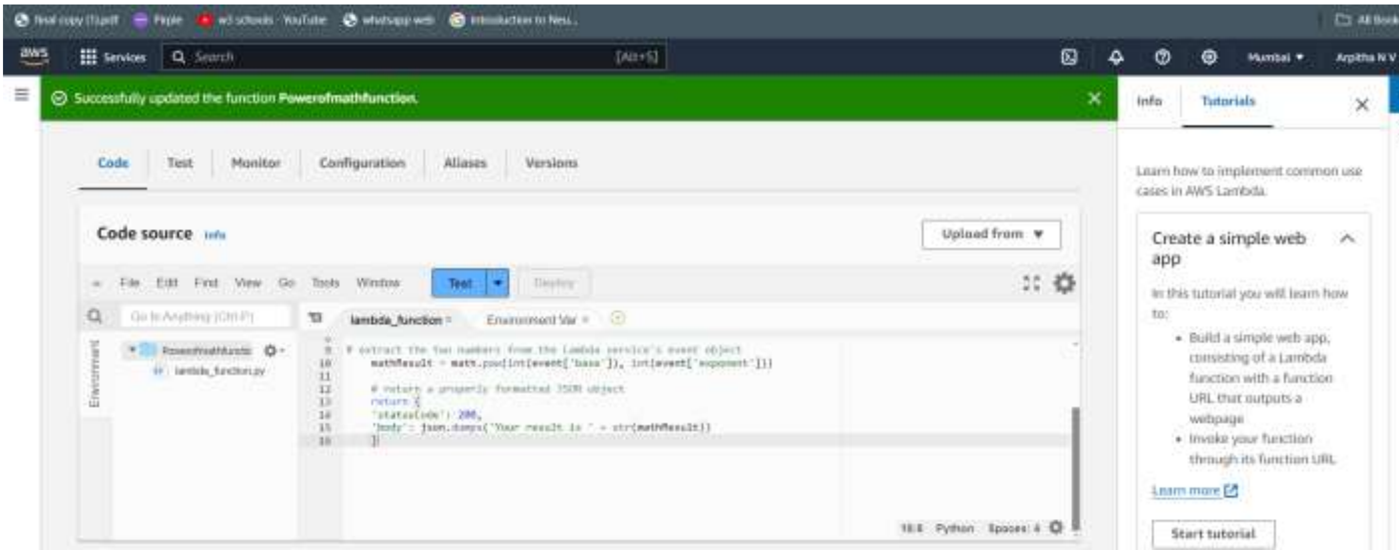
mathResult = math.pow(int(event['base']), int(event['exponent'])):

This line extracts the values of base and exponent from the event object, converts them to integers, and calculates the power (base raised to the power of exponent) using math.pow.

Copy and Paste the code in Code Source Tab,

Deploy – Test – Drop down – Configure Test Event – Create a new event

Event Name= Powerofmathevent



Successfully updated the function Powerofmathfunction.

```
# extract the two numbers from the Lambda service's event object
    mathResult = math.pow(int(event['base']), int(event['exponent']))

    # return a properly formatted JSON object
    return {
    'statusCode': 200,
    'body': json.dumps('Your result is ' + str(mathResult))
    }
```

Step 4: Deploy an API Gateway to Invoke a Math functionality

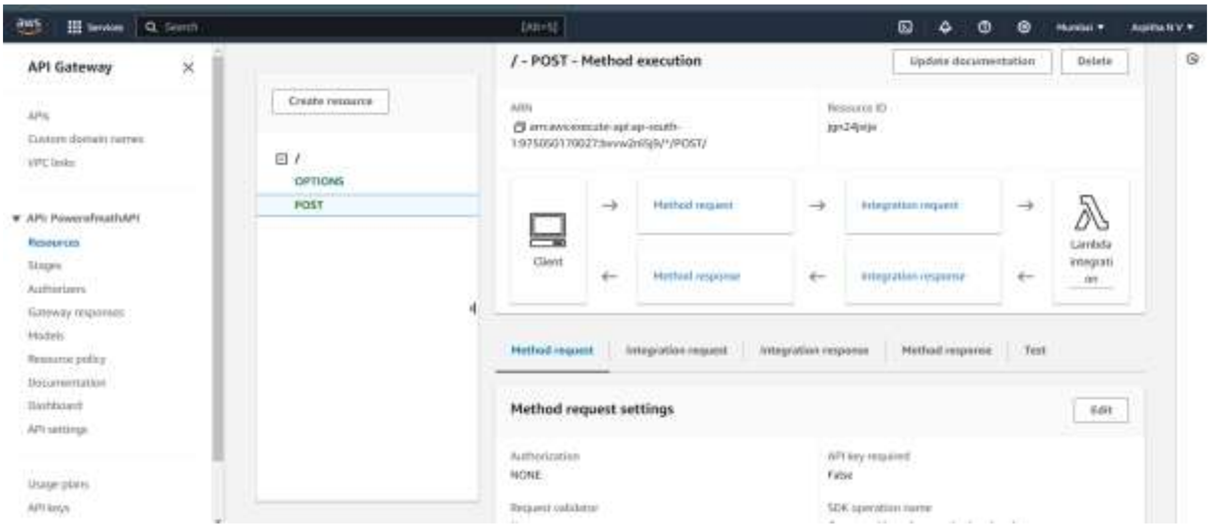
Open Amazon Console and Navigate to API Gateway

Create API – Rest API – Buid – Rest – New API

API Name= PowerofmathAPI

Actions – POST

SAVE



/ - POST - Method execution

Method request settings

Authorization: NONE

API key required: False

Actions – Enable CORS (Cross- Origin Resource Source)

Deplyment Stage – New stage

Stage Name – Dev

DEPLOY

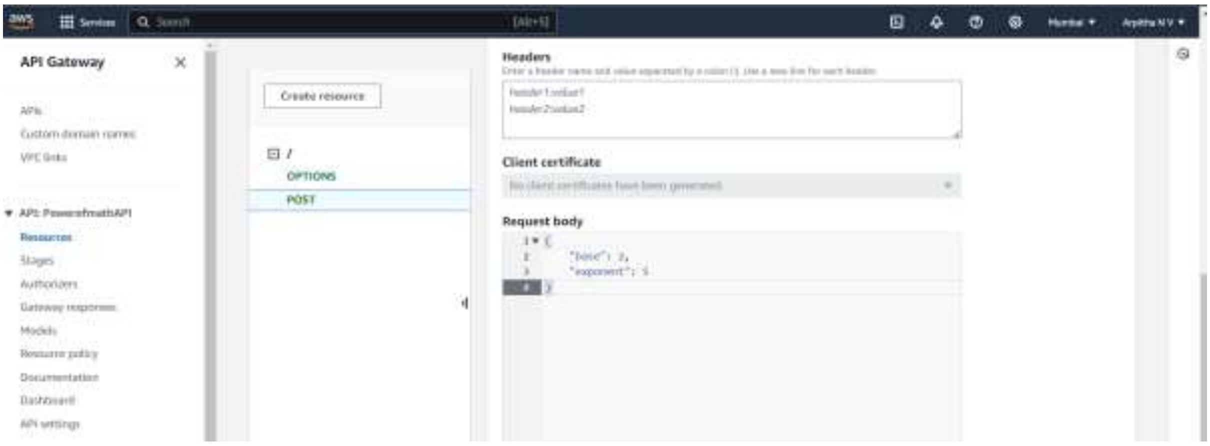
Copy the URL that pops up on INVOKE URL

https://bvvw2nl5j9.execute-api.ap-south-1.amazonaws.com/dev

Actions – Resources – Post Method Tests – Perform the test in Request body Tab

Eg:

Test:



Headers

Enter a header name and value separated by a colon (:). Use a new line for each header.

header1:value1
header2:value2

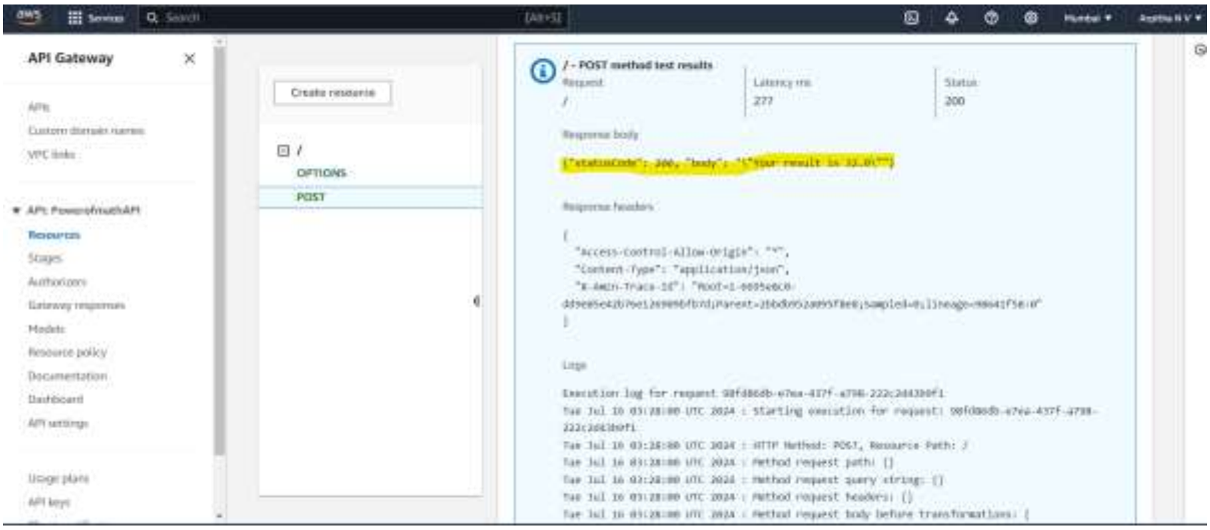
Client certificate

No client certificates have been generated

Request body

```
{
    "base": 2,
    "exponent": 5
}
```

Result:



/ - POST method test results

Request	Latency ms	Status
/	277	200

Response body

```
{"statusCode": 200, "body": "\"Your result is 32.0\""}
```

Response headers

```
{
  "Access-Control-Allow-Origin": "*",
  "Content-Type": "application/json",
  "X-Amzn-Trace-Id": "Root=1-6695e6c0-
dd9e05e42b76e12e9096fb7d;Parent=2bbd0952a095f8e0;Sampled=0;lineage=f8641f5e:0"
}
```

Logs

```
Execution log for request 98fd86db-e7ea-437f-a798-222c2d4398f1
Tue Jul 16 03:28:00 UTC 2024 : Starting execution for request: 98fd86db-e7ea-437f-a798-
222c2d4398f1
Tue Jul 16 03:28:00 UTC 2024 : HTTP Method: POST, Resource Path: /
Tue Jul 16 03:28:00 UTC 2024 : Method request path: {}
Tue Jul 16 03:28:00 UTC 2024 : Method request query string: {}
Tue Jul 16 03:28:00 UTC 2024 : Method request headers: {}
Tue Jul 16 03:28:00 UTC 2024 : Method request body before transformations: {
```

Step 5: Use Dynamo DB to store/return Math Results.

Go to the console page and navigate to Amazon Dynamo Db

Dynamo DB dashboard -- Create Table

Table Name= Powerofmathdatabase

Partition key = ID

CREATE TABLE

Click o the name of the created table in the tables option.

Under General Information -- Additional Info -- copy the ARN provided.

Eg: arn:aws:dynamodb:ap-south-1:975050170027:table/Powerofmathdatabase

Step 6: Setup Permissions to Lambda Functions to access the Dynamo db Table Using IAM.

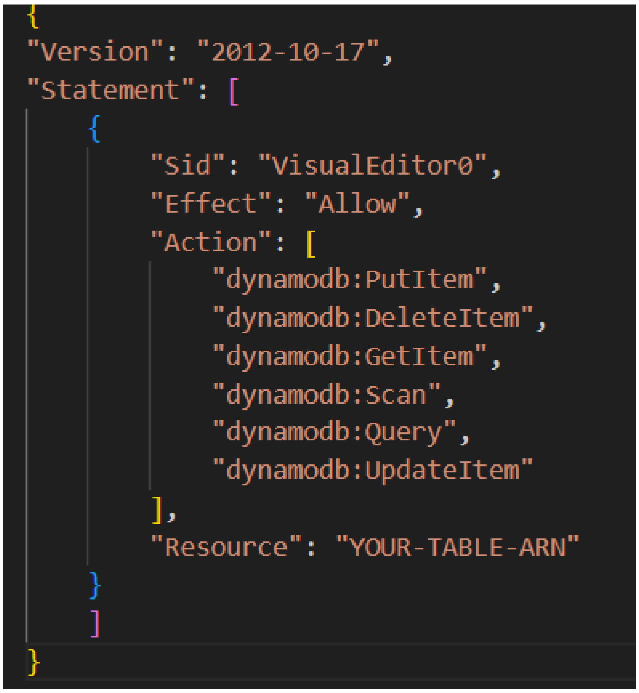
Navigate to Lambda Functions Tab

Configurations – Permissions – Click on role Name link –

Automatically navigates to IAM Tab,

Add permissions -- Drop Down – Create inline Ploicy – JASON

Write the JSON Code to provide a policy grants specific permissions to interact with an Amazon DynamoDB table.



```
{
"Version": "2012-10-17",
"Statement": [
    {
        "Sid": "VisualEditor0",
        "Effect": "Allow",
        "Action": [
            "dynamodb:PutItem",
            "dynamodb:DeleteItem",
            "dynamodb:GetItem",
            "dynamodb:Scan",
            "dynamodb:Query",
            "dynamodb:UpdateItem"
        ],
        "Resource": "YOUR-TABLE-ARN"
    }
    ]
}
```

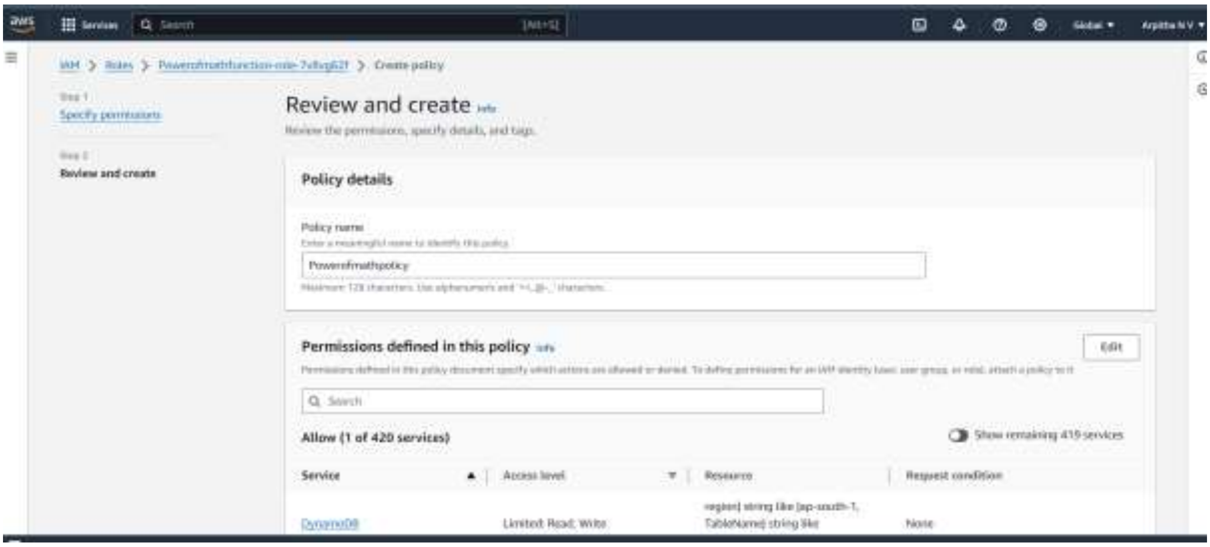
Also paste the specific ARN that was Copies from API Gateway

This IAM policy grants permissions to perform the specified actions on a particular DynamoDB table. It allows putting, deleting, getting, scanning, querying, and updating items within that table.

Click on Review Policy

Policy Name =Poweofmathpolicy

CREATE POLICY



IAM > Roles > Powerofmathfunction-role-7v8vg62f > Create policy

Step 1
Specify permissions

Step 2
Review and create

Review and create

Review the permissions, specify details, and tags.

Policy details

Policy name
Enter a meaningful name to identify this policy.

Powerofmathpolicy

Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.

Permissions defined in this policy

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it.

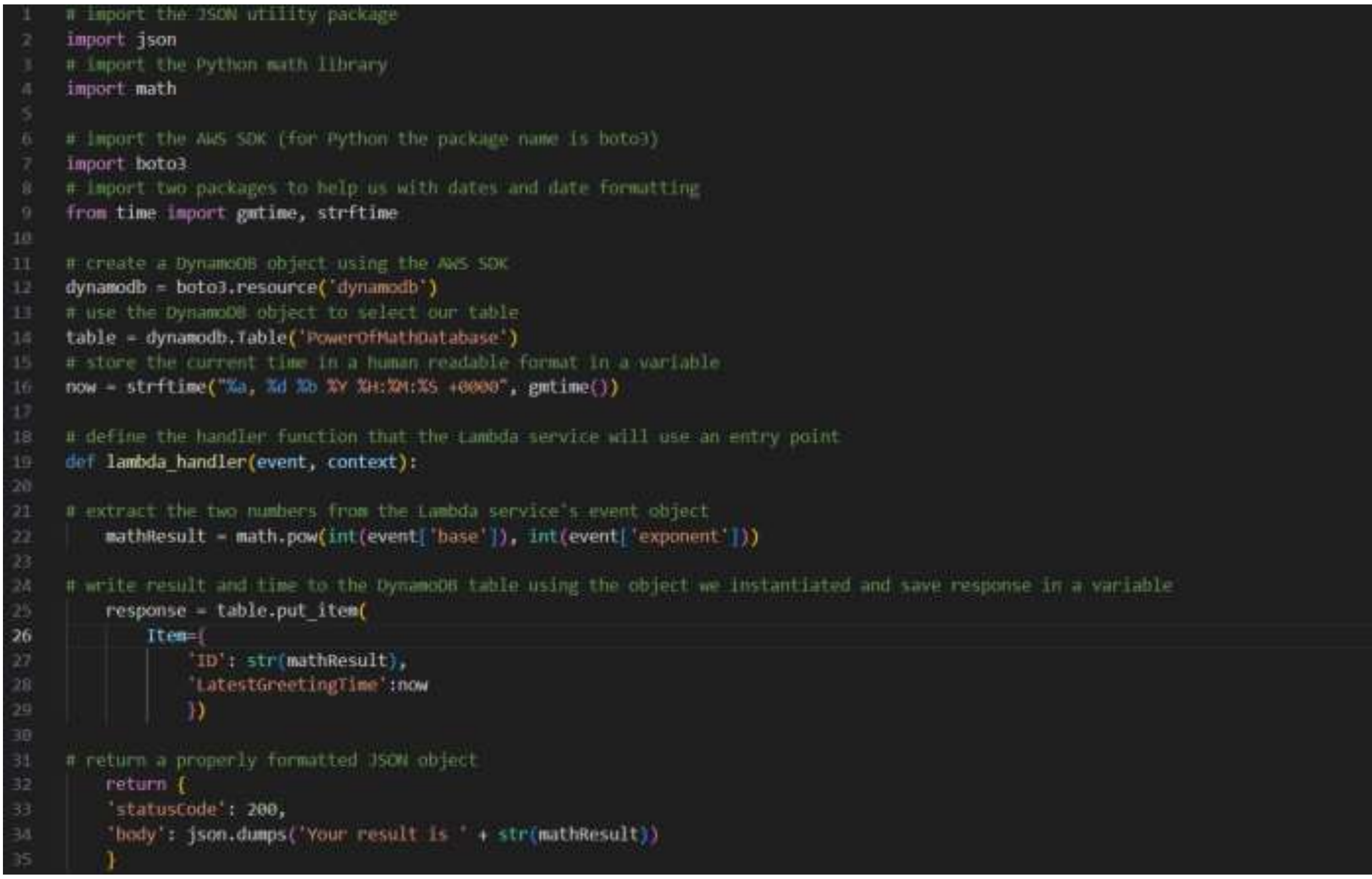
Allow (1 of 420 services)

Service	Access level	Resource	Request condition
DynamoDB	Limited: Read, Write	region| string like |ap-south-1, TableName| string like	None

Come back to Lambda Functions Tab,

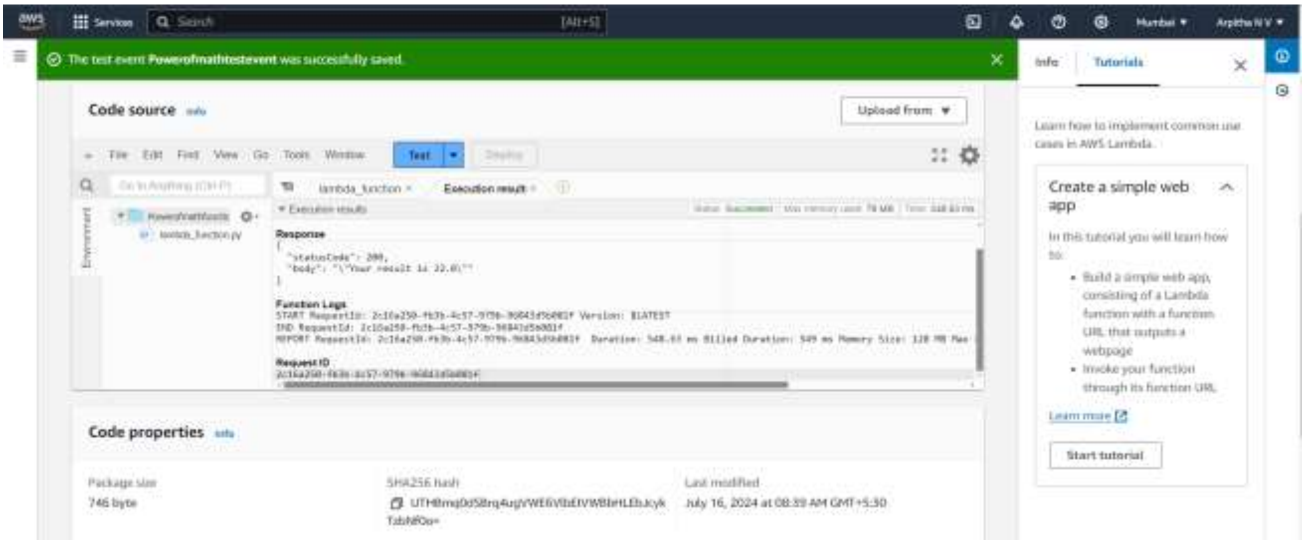
Code,

Write a Python code , where an AWS Lambda function that performs a mathematical power calculation and stores the result along with the current timestamp in an Amazon DynamoDB table.



```python
# import the JSON utility package
import json
# import the Python math library
import math

# import the AWS SDK (for Python the package name is boto3)
import boto3
# import two packages to help us with dates and date formatting
from time import gmtime, strftime

# create a DynamoDB object using the AWS SDK
dynamodb = boto3.resource('dynamodb')
# use the DynamoDB object to select our table
table = dynamodb.Table('PowerOfMathDatabase')
# store the current time in a human readable format in a variable
now = strftime("%a, %d %b %Y %H:%M:%S +0000", gmtime())

# define the handler function that the Lambda service will use an entry point
def lambda_handler(event, context):

# extract the two numbers from the Lambda service's event object
    mathResult = math.pow(int(event['base']), int(event['exponent']))

# write result and time to the DynamoDB table using the object we instantiated and save response in a variable
    response = table.put_item(
        Item={
            'ID': str(mathResult),
            'LatestGreetingTime':now
            })

# return a properly formatted JSON object
    return {
    'statusCode': 200,
    'body': json.dumps('Your result is ' + str(mathResult))
    }
```

Deply and Test the code,



The test event **Powerofmathtestevent** was successfully saved.

Code source

```
Response
{
  "statusCode": 200,
  "body": "\"Your result is 22.0\""
}

Function Logs
START RequestId: 2c16a250-fb3b-4c57-979b-9b843d5b081f Version: $LATEST
END RequestId: 2c16a250-fb3b-4c57-979b-9b843d5b081f
REPORT RequestId: 2c16a250-fb3b-4c57-979b-9b843d5b081f  Duration: 348.83 ms  Billed Duration: 349 ms  Memory Size: 128 MB  Max

Request ID
2c16a250-fb3b-4c57-979b-9b843d5b081f
```

Code properties

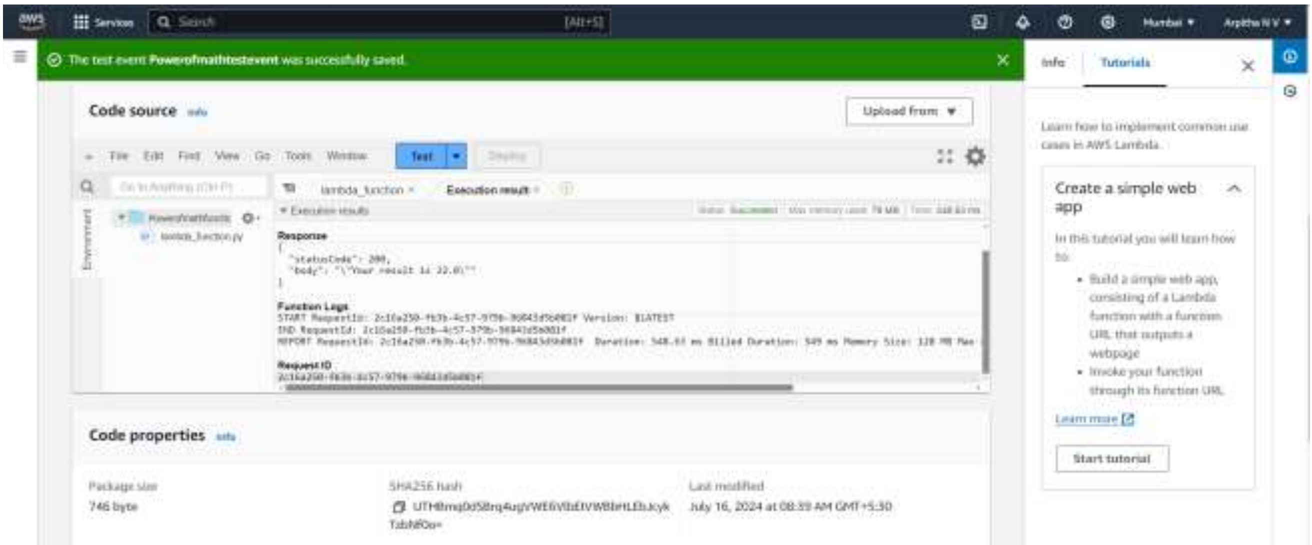
Package size	SHA256 hash	Last modified
746 byte	UTHBmqDd58rq4ugVWE6VbEIVWBIeHLEbJcyk TzbNfOo=	July 16, 2024 at 08:39 AM GMT+5:30

Learn how to implement common use cases in AWS Lambda.

Create a simple web app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL.



The test event **Powerofmathtestevent** was successfully saved.

Code source

```
Response
{
  "statusCode": 200,
  "body": "\"Your result is 22.0\""
}

Function Logs
START RequestId: 2c16a250-fb3b-4c57-979b-9b843d5b081f Version: $LATEST
END RequestId: 2c16a250-fb3b-4c57-979b-9b843d5b081f
REPORT RequestId: 2c16a250-fb3b-4c57-979b-9b843d5b081f  Duration: 348.83 ms  Billed Duration: 349 ms  Memory Size: 128 MB  Max

Request ID
2c16a250-fb3b-4c57-979b-9b843d5b081f
```

Code properties

Package size 746 byte
SHA256 hash UTHBmqDd58rq4ugVWE6VbEIVWBIeHLEbJcyk TzbNfOo=
Last modified July 16, 2024 at 08:39 AM GMT+5:30

Learn how to implement common use cases in AWS Lambda.

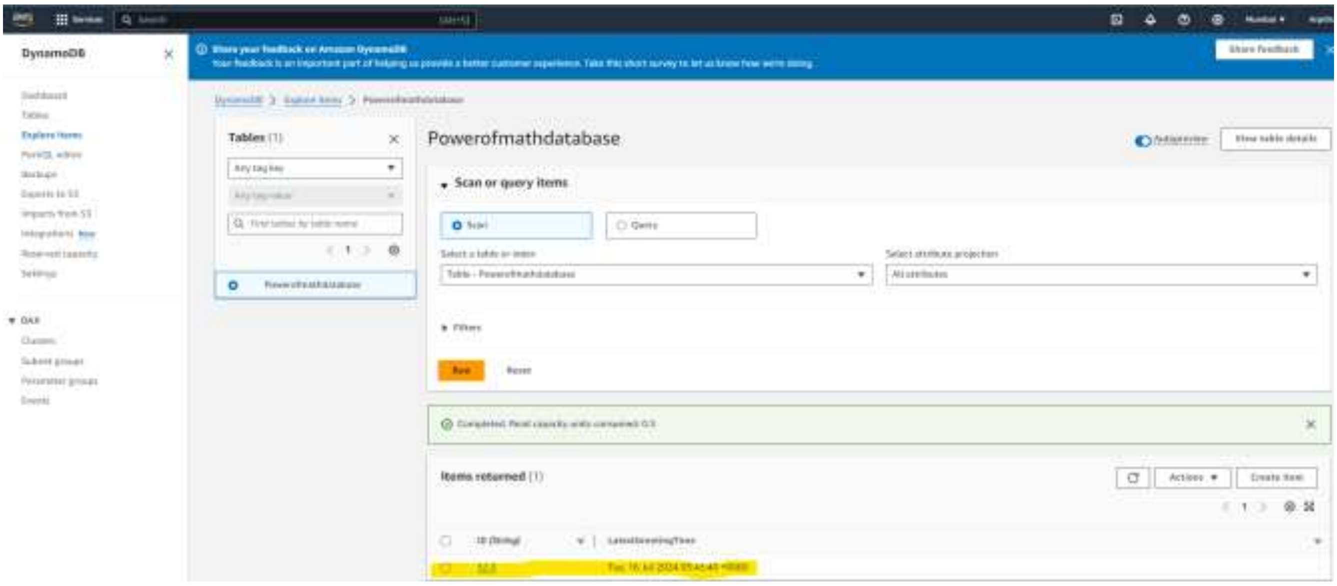
Create a simple web app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL.

We can come back to Dynamo db ttab and check if the Results are save in the Table.

Click on Explore Table,



Powerofmathdatabase

Completed. Read capacity units consumed: 0.5

Items returned (1)

ID (String)	LatestGreetingTime
32.0	Tue, 16 Jul 2024 05:45:48 +0000

Step 7:

Write an HTML code to create a simple web interface that allows users to input a base and an exponent, then calculates the power of the base raised to the exponent by making an API call to an AWS Lambda function.

Replace "YOUR API GATEWAY ENDPOINT" with the actual endpoint of your AWS API Gateway that invokes the Lambda function

Eg:

ARN URL: arn:aws:dynamodb:ap-south-1:975050170027:table/Powerofmathdatabase

CODE

```
<!DOCTYPE html>

<html>

<head>

    <meta charset="UTF-8">

    <title>To the Power of Math!</title>

    <!-- Styling for the client UI -->

    <style>

    h1 {
```

```
    color: #FFFFFF;
    font-family: system-ui;
                      margin-left: 20px;
    }
          body {
    background-color: #222629;
    }
label {
    color: #86C232;
    font-family: system-ui;
    font-size: 20px;
    margin-left: 20px;
                      margin-top: 20px;
    }
 button {
    background-color: #86C232;
                      border-color: #86C232;
                      color: #FFFFFF;
    font-family: system-ui;
    font-size: 20px;
                      font-weight: bold;
    margin-left: 30px;
                      margin-top: 20px;
                      width: 140px;
    }
           input {
    color: #222629;
    font-family: system-ui;
    font-size: 20px;
    margin-left: 10px;
             margin-top: 20px;
             width: 100px;
    }
  </style>
  <script>
    // callAPI function that takes the base and exponent numbers as parameters
    var callAPI = (base,exponent)=>{
      // instantiate a headers object
      var myHeaders = new Headers();
      // add content type header to object
      myHeaders.append("Content-Type", "application/json");
      // using built in JSON utility package turn object to string and store in a variable
      var raw = JSON.stringify({"base":base,"exponent":exponent});
      // create a JSON object with parameters for API call and store in a variable
      var requestOptions = {
        method: 'POST',
        headers: myHeaders,
        body: raw,
        redirect: 'follow'
      };
      // make API call with parameters and use promises to get response
      fetch("https://bvvw2nl5j9.execute-api.ap-south-1.amazonaws.com/dev",
requestOptions)
      .then(response => response.text())
      .then(result => alert(JSON.parse(result).body))
      .catch(error => console.log('error', error));
    }
  </script>
</head>
```

```
<body>
  <h1>TO THE POWER OF MATH!</h1>
      <form>
    <label>Base number:</label>
    <input type="text" id="base">
    <label>...to the power of:</label>
    <input type="text" id="exponent">
    <!-- set button onClick method to call function we defined passing input values as parameters -->
    <button type="button" onclick="callAPI(document.getElementById('base').value,document.getElementById('exponent').value)">CALCULATE</button>
  </form>
</body>
</html>
```

index.html

Displaying index.html.

Save the file as, index.html Zip it.

Amplify Tab –

Drag a Zip File that was create earlier, Save and Deploy

Once the Deployment is Successful.

Copy the URL link provided in the Domain and paste in a new tab,

THE Web page Opens,

Provide the Base: 2

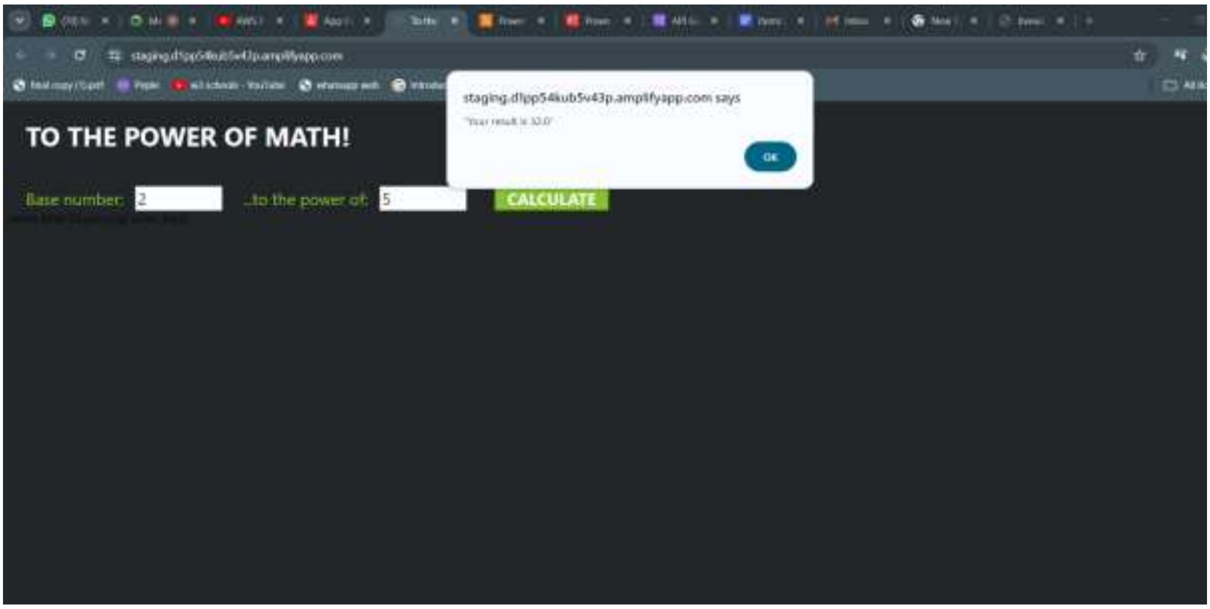
The Power: 4

Click on CALCULATE

TAAADDDAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

The answer Pop up!!!

2*2*2*2*2 = 32



staging.d1pp54kub5v43p.amplifyapp.com says

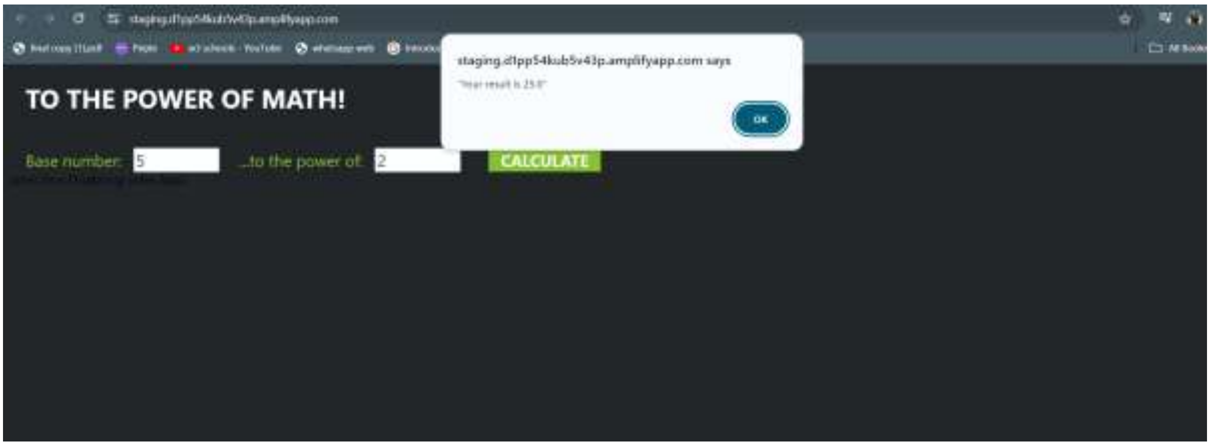
"Your result is 32.0"

TO THE POWER OF MATH!

Base number: 2 ...to the power of: 5

RIGHT????

AGAIN



staging.d1pp54kub5v43p.amplifyapp.com says

"Your result is 25.0"

TO THE POWER OF MATH!

Base number: 5 ...to the power of: 2

THANK YOU!

DON'T FOGET TO TERMINATE THE RESOURCES AND DELETE ONCE DONE!